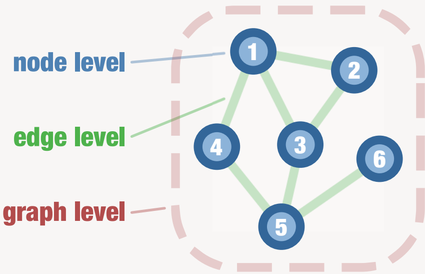
Cheatsheet: GNNs

Graph Neural Networks and Transformers: Neural Networks for Sets

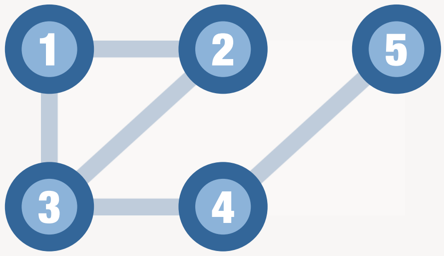
target levels for representation

- Node level representation
  Example: Social network bot / troll detection
- Edge level representation
  Example: Social network missing link prediction
- Graph level representation
  Example: Molecular property prediction (e.g., toxicity, solubility coeffcient)

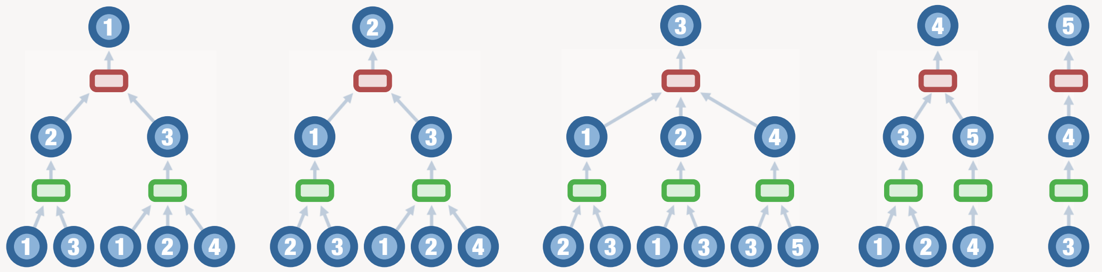


representational graph

- Refers to graph representation of the problem at hand

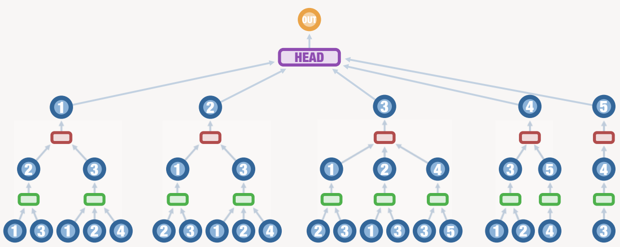


GNN compute graph



- Derived (unrolled) from representational graph above
- Shown here: 2 hops compute graph (i.e., compute graph for a 2-layer GNN)
- Green: Layer 1 (message compute & aggregation)
- Red: Layer 2 (message compute & aggregation)
- Same color (red / green) = shared weights
- Representational graph can be derived from compute graph and vice versa

getting to the target level



- In most cases, core GNNs produce node-level representations
- Edge-level representation: aggregating representations of adjacent nodes
- Graph-level representation: aggregating representations of all nodes in the graph
- Example above: graph level task

Head for edge representations:

$$y_{uv} = Head_{edge}\left\{h_u^{(l)}, h_v^{(l)}\right\}$$

Examples:

$$y_{uv} = Linear(Concat(h_u^{(l)}, h_v^{(l)}))$$
$$y_{uv} = h_u^{(l)} + h_v(l)$$
$$y_{uv} = dot(h_u^{(l)}, h_v^{(l)})$$

Head for graph representations:

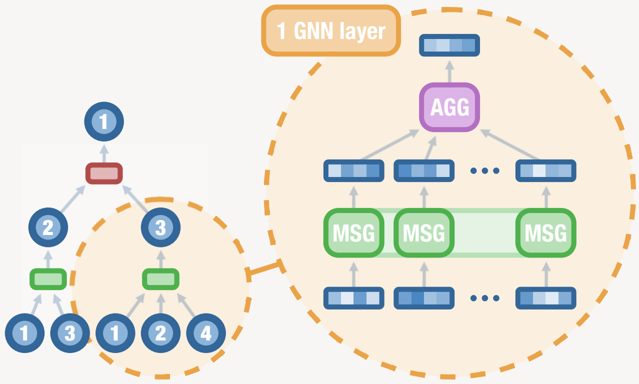
$$y_G = Head_{graph}\left\{(h_v^{(l)}) \in R^d, \forall v \in G)\right\}$$

Cheatsheet: GNNs

Graph Neural Networks and Transformers: Neural Networks for Sets

message computation

- Node features get transformed by a function MSG(), typically a neural network (MLP)
- Input: fixed-length feature vector (concatenation or sum of multiple embeddings)
- Output: Message (fixed-length vector)
- Function is applied to each input vector individually



One GNN layer:

$$h_v^{(l)} = AGG^{(l)}\left(\left\{MSG^{(l)}\left(h_u^{(l-1)}\right), u \in N(v)\right\}\right)$$

Alternatives for $MSG(\cdot)$:

$$MSG^{(l)}\left(h_u^{(l-1)}\right) = m_u^{(l)} = W^{(l)} h_u^{(l-1)}$$
$$MSG^{(l)}\left(h_u^{(l-1)}\right) = m_u^{(l)} = MLP(h_u^{(l-1)})$$

Alternatives for $AGG(\cdot)$:

$$AGG^{(l)}\left(\left\{m_u^{(l)}, u \in N(v)\right\}\right) = Sum\left(\left\{m_u^{(l)}, u \in N(v)\right\}\right) = \sum_{u \in N(v)} m_u^{(l)}$$

$$AGG^{(l)}(\cdot) = Mean(\cdot)$$
$$AGG^{(l)}(\cdot) = Max(\cdot)$$
$$AGG^{(l)}(\cdot) = MLP(Sum(\cdot))$$

Optional extensions:

$$h_v^{(l)} \leftarrow AGG^{(l)}\left(\left\{MSG^{(l)}\left(h_u^{(l-1)}\right), u \in N(v)\right\}\right)$$
$$h_v^{(l)} \leftarrow CONCAT\left(h_v^{(l)}, MSG_t^{(l)}\left(h_v^{(l-1)}\right)\right)$$
$$h_v^{(l)} \leftarrow \sigma\left(W^l \cdot h_v^{(l)}\right)$$
$$h_v^{(l)} \leftarrow \frac{h_v^{(l)}}{\|h_v^{(l)}\|_2}$$

message aggregation

- Multiple messages get aggregated into a single vector by a function AGG()
- Input: N messages
- Output: Fixed-length vector (typically of same size)
- Requirement: Permutation invariance (no concatenation!)
- Examples: Sum(), Mean(), Max()
- Optimal: MLP(Sum(message)) with message = MLP(feature)

layer connectivity

- Each GNN-layer increases "receptive field" by one hop
- GNNs typically have 2-3 layers (more is often not helpful)
- Instead:
  - Increase the size / number of layers of the networks within the GNN-layers
  - Pre-process raw features with deep networks
  - Add skip-connections